ALGORITMA PENJADWALAN *REQUEST* PADA JARINGAN RADIO SINKRON

Nola Marina

Fakultas Teknologi Industri, Universitas Gunadarma
Jl. Margonda Raya no. 100, Depok 16424, Jawa Barat
nolamarina@staff.gunadarma.ac.id

Abstrak

Teknologi jaringan nirkabel bertujuan untuk memudahkan pertukaran informasi antar perangkat. Jaringan Radio Sinkron (JRS) adalah jaringan nirkabel yang menghubungan perangkat (simpul) untuk bertukar data melalui media gelombang radio dan bekerja secara sinkron. Penggunaan medium radio pada JRS menimbulkan masalah berupa mudahnya terjadi interferensi data. Untuk menjamin agar semua pengiriman data berhasil diterima oleh simpul tujuan, diperlukan suatu penjadwalan pengiriman data (penjadwalan request) berdasarkan kendala pada model interferensi. Penjadwalan request dikatakan baik jika menggunakan total slot waktu minimal dan menjamin tidak terjadi interferensi. Urutan request yang dijadwalkan dan teknik pengalokasian slot waktu pada setiap request sangat berpengaruh terhadap total slot waktu yang digunakan hingga semua pengiriman data selesai. Pada penelitian ini dipelajari algoritma penjadwalan request pada JRS topologi rantai yang menggunakan pendekatan greedy dalam mencari urutan request terbaik dan mengalokasikan slot waktu pada setiap request. Algoritma ini diimplementasikan pada 8 data dan diperoleh bahwa algoritma dengan pendekatan greedy memberikan total slot waktu 10.18 % lebih baik dibandingkan algoritma tanpa pengurutan (trivial).

***Kata Kunci**: Jaringan, Penjadwalan, Radio, Rantai.*

REQUEST SCHEDULING ALGORITHM ON THE SYNCHRONOUS RADIO NETWORK

Abstract

Wireless network technology aims to facilitate the exchange of information between devices. Synchronous Radio Network (SRN) is a wireless network that connects devices (nodes) to exchange data through radio wave media and work in sync. The use of radio medium in SRN cause problems in the form of easy interference of data. To ensure that all data request are successfully received by the destination node, a data scheduling (request scheduling) is required based on constraints on the interference model. The sequence of request and time slot allocation technique on each request greatly affects to the total time slot used until all data transmissions are completed. In this study we studied the request scheduling algorithm on SRN chain topology using the greedy approach in finding the best sequence and allocating time slot on each request. This algorithm is implemented on 8 data and it is found that algorithm with greedy approach gives total slot time of 10.18% better than algorithm without sequencing (trivial).

***Keywords**: Chain, Network, Radio, Scheduling.*

PENDAHULUAN

Jaringan terdiri atas stasiun (*node*) dan konektivitasnya (*link*). Jaringan nirkabel adalah jaringan yang menghubungan perangkat-perangkat (simpul) untuk bertukar data tanpa menggunakan media kabel, melainkan media berupa gelombang radio, infra merah atau gelombang mikro. Pemanfaatan jaringan nirkabel terutama terdapat pada bidang telekomunikasi, seperti jaringan paket radio, jaringan telepon seluler, jaringan area lokal nirkabel, jaringan satelit, dan jaringan sensor.

Jaringan nirkabel yang menggunakan media gelombang radio elektromagnetik disebut jaringan radio. Jaringan radio merupakan kumpulan dari simpul pengirim dan penerima yang berkomunikasi antara satu dengan lainnya melalui *link multihop*. Transmisi data adalah pengiriman data dari simpul pengirim ke simpul penerima. Permintaan transmisi suatu data dengan rute tertentu disebut dengan *request*.

Penggunaan medium radio pada jaringan radio menyebabkan adanya karakteristik berikut: 1) ketika simpul mengirim data, semua simpul dalam jangkauan transmisinya dapat menerima sinyal transmisi. 2) Data yang dikirim dapat mencapai simpul lebih dari satu hop dari simpul asal. 3) Karena semua simpul memiliki frekuensi yang sama, tabrakan data dapat terjadi jika dua atau lebih simpul tetangga mengirim data secara bersamaan [1].

Pada penelitian ini jaringan radio diasumsikan bersifat *omnidirectional*, sinkron, dan *half duplex*. Beberapa model interferensi diberikan pada penelitian [1, 2, 3, 4]. Pada penelitian ini, model interferensi yang digunakan adalah berdasarkan [1] dan [4], yaitu interferensi terjadi jika: 1) suatu simpul mengirim lebih dari satu data dalam satu waktu, 2) suatu simpul mengirim dan menerima data secara bersamaan, 3) terdapat dua atau lebih simpul tetangga yang mengirimkan data dalam waktu yang sama.

Untuk menjamin agar data yang dikirimkan berhasil diterima dengan baik dan untuk mengetahui perkiraan waktu yang dibutuhkan hingga data berhasil diterima, diperlukan suatu penjadwalan *request* (pengiriman data). Penjadwalan *request* merupakan masalah yang sangat penting dalam jaringan radio.

Pada penelitian ini didefinisikan masalah penjadwalan pada Jaringan Radio Sinkron (JRS) adalah bagaimana pengalokasian slot waktu pada sejumlah *request* dengan rute masing-masing (dari simpul sumber ke simpul tujuan) dalam suatu jaringan sedemikian sehingga jadwal valid, tidak ada konflik dan total slot waktu yang digunakan minimal.

Penjadwalan yang baik dipengaruhi oleh urutan kumpulan *request* yang dijadwalkan dan cara mengalokasikan slot waktu untuk setiap *request*. Jika urutan *request* dalam proses mengalokasikan slot waktu berbeda maka total slot waktu yang digunakan dapat berbeda pula. Berdasarkan hal ini, salah satu bagian dari masalah penjawalan *request* pada JRS adalah bagaimana menemukan urutan *request* yang menghasilkan penjadwalan dengan total slot waktu minimal dan bagaimana teknik mengalokasikan slot waktu pada setiap *request*. Pada penelitian ini dipelajari algoritma penjadwalan *request* pada JRS topologi rantai yang memperhatikan masalah tesebut.

METODE PENELITIAN

Jaringan radio umumnya direpresentasikan dalam bentuk graf. Pendekatan graf sangat penting dalam menyelesaikan masalah penjadwalan *request* pada JRS dan menentukan kompleksitas masalahnya.

Suatu jaringan radio direpresentasikan oleh graf tak berarah G dimana

himpunan *vertex* $V(G)$ merepresentasikan himpunan simpul dalam jaringan. Sedangkann sisi $e = \{u, v\} \in E(G)$ menggambarkan bahwa simpul u berkomunikasi langsung dengan simpul v dan sebaliknya.

Request $\boldsymbol{r}$ adalah pasangan $(s,t), dimana\ s,t \in V(G)$, s adalah simpul asal dan t adalah simpul tujuan. **Path** panjang k pada graf G adalah urutan $(v_0, v_1, \ldots, v_k)$ dimana $v_i \in V(G)$ untuk setiap $i \in [0,k]$ sedemikian sehingga edge (v_i, v_{i+1}) ada di $E(G)$ untuk setiap $i \in [0, k-1]$, dan setiap edge adalah berbeda. Path merepresentasikan jalan/rute komunikasi data pada jaringan. Misalkan P adalah **fungsi routing** pada R yang berasosiasi dengan setiap $r = (s,t) \in R$. Path $\boldsymbol{P(r)}$ **atau** $\boldsymbol{P_r}$ di G dimulai dari simpul asal s dan berakhir pada simpul tujuan t.

Diberikan graf G, kumpulan *request* R, dan fungsi routing P, maka didefinisikan ***date assignment*** d adalah fungsi dari $r = (s,t) \in R$ dan $x \in P_r$ $(x \neq t)$ yang mengembalikan nilai $\boldsymbol{d(r,x)}$**.** Nilai $d(r,x)$ ini berkorespondensi dengan suatu slot waktu, yang artinya x mengirim data dari r ke hop berikutnya selama slot waktu tersebut.

Suatu *assignment* dikatakan **valid** jika untuk setiap *request* $r = (x_0, x_k)$ dengan fungsi routing $P_r = (x_0, x_1, \ldots, x_k)$, memenuhi syarat: $d(r, x_0) < d(r, x_1) < \cdots . < d(r, x_{k-1}) < d(r, x_{k-1})$.

Suatu assignment dikatakan **bebas konflik** jika untuk setiap $d(r, x_i) \neq d(r', y_j)$dan $r \neq r'$, memenuhi ketiga syarat berikut:

1. Setiap transmisi hanya boleh mengirim satu data :
$$x_i \neq y_j$$
2. Suatu simpul tidak dapat mengirim dan menerima data secara bersamaan dalam satu slot waktu :
$x_{i+1} \neq y_j$ dan $x_i \neq y_{j+1}$
3. Tidak boleh terjadi tabrakan/ collide:
$\{x_i, y_{j+1}\} \notin E(G) \wedge \{y_j, x_{i+1}\} \notin E(G)$

Total waktu suatu penjadwalan adalah jumlah slot waktu yang digunakan agar semua *request* sampai ke simpul tujuan dinotasikan dengan:
$$\max(d) = \max_{(r \in R, x \in V(G)} d(r,x)$$
yaitu slot waktu terbesar dari semua slot yang telah dialokasikan.

Masalah penjadwalan *request* adalah masalah optimasi yang termasuk NP-*hard*. NP-*hard* adalah kelas masalah yang sulit secara komputasi [5]. Algoritma *greedy* dikenal sebagai algoritma sederhana yang efektif untuk menyelesaikan masalah optimasi. Algoritma *greedy* adalah algoritma yang memecahkan masalah langkah per langkah, pada setiap langkah mengambil pilihan yang terbaik yang dapat diperoleh pada saat itu tanpa memperhatikan konsekuensi ke depan dan berharap bahwa dengan memilih optimum lokal pada setiap langkah akan berakhir dengan optimum global.

Pada penelitian ini dibangun algoritma penjadwalan untuk penjadwalan *request* pada JRS topologi rantai menggunakan pendekatan greedy. Pendekatan *greedy* digunakan untuk menentukan urutan *request* yang dijadwalkan dan juga dalam mengalokasikan slot waktu pada setiap *request*.

HASIL DAN PEMBAHASAN

Diketahui sebuah jaringan berupa sebuah graf tak berarah berbentuk rantai, dengan C=$[1,2,..n]$ dan kumpulan *request* $R = \{r_i = (s_i, t_i) | 1 \leq i \leq m\}$. Banyak *vertex* adalah n dan banyak *request* adalah m. Model eksperimental JRS topologi rantai direpresentasikan dalam bentuk matriks $D_{mxn-1}(i,j)$, dengan $i =$ *request* ke -i dan $j =$ *edge* $(j, j+1)$, $M = [1,2,..,m], N = [1,2,..,n]$,$D(i,j)$= slot waktu pengiriman *request*- i ke simpul berikutnya.

Kendala yang harus dipenuhi oleh penjadwalan diberikan oleh model berikut:

a. Jadwal valid

Secara formal, suatu jadwal dikatakan valid jika setiap *request* $r = (x_0, x_k)$ dengan fungsi routing $P_r = (x_0, x_1, \dots, x_k)$, memenuhi syarat: $d(r, x_0) < d(r, x_1) < \cdots < d(r, x_{k-1}) < d(r, x_{k-1})$. Hal ini dapat direpresentasikan dengan syarat berikut pada matriks $D(i,j)$:

1. $D(i,j) < D(i,j+1) < \cdots < D(i,j+k-1)$, , untuk i dimana $s_i < t_i$
2. $D(i,j) < D(i,j-1) < \cdots < D(i,j+k-1)$, untuk i dengan $s_i > t_i$

b. Jadwal bebas konflik

Suatu jadwal dikatakan bebas konflik jika untuk setiap $d(r, x_i) = d(r', y_j)$ dan $r \neq r'$, memenuhi ketiga syarat berikut:

1. $x_i \neq y_j$
2. $x_{i+1} \neq y_j$ dan $x_i \neq y_{j+1}$
3. $x_{i+1} \neq y_{i+1} \ \wedge \ \{x_i, y_{j+1}\} \notin E(G) \ \wedge \ \{y_j, x_{i+1}\} \notin E(G)$

Ketiga syarat jadwal bebas konflik di atas dapat direpresentasikan dengan syarat berikut pada matriks $D(i,j)$ model eksperimental:

Untuk r_k yang searah dengan r_i, $\forall i, k \neq i$ dengan $i \in M$ dan $k \in N$.

1. $D(i,j) \neq D(k,j)$, syarat bebas konflik ke-1
2. $D(i,j) \neq D(k,j+1)$, syarat bebas konflik ke-2
3. $D(i,j) \neq D(k,j+2)$, syarat bebas konflik ke-3

Untuk r_k yang beda arah dengan r_i, $\forall i, k \neq i$ dengan $i \in M$ dan $k \in N$.

1. $D(i,j) \neq D(k,j+1)$, syarat bebas konflik ke-1
2. $D(i,j) \neq D(k,j)$, syarat bebas konflik ke-2

4. Jika pada D_i_temp terdapat slot waktu yang tidak bebas konflik, maka ulangi penjadwalan $D(i, x_0)$ dengan
3. $D(i,j) \neq D(k,j+1)$, syarat bebas konflik ke-3

Syarat diatas dapat digabungkan penulisannya dan menjadi syarat bebas konflik pada model eksperimental, yaitu:

1. $D(i,j) \neq [D(k,j-2), D(k,j-1), D(k,j), D(k,j+1), D(k,j+2)]$,

untuk r_k yang searah dengan r_i , $k \neq i$ dan

2. $D(i,j) \neq [D(k,j-1), D(k,j), D(k,j+1)]$,

untuk r_k yang tidak searah dengan r_i, $k \neq i$, dengan $i \in M$ dan $k \in N$.

Algoritma Penjadwalan dengan Pendekatan *Greedy*

Algoritma untuk penjadwalan *request* pada JRS topologi rantai menggunakan pendekatan *greedy* untuk mengalokasikan slot waktu pada setiap *request* dan menentukan urutan *request* yang dijadwalkan.

Cara mengalokasikan slot waktu dengan pendekatan *greedy* dilakukan dengan menjadwalkan satu *request* sesegera mungkin dengan slot waktu terkecil yang *available*, terus-menerus hingga sampai ke simpul tujuan, baru kemudian menjadwalkan *request* berikutnya.

Berikut adalah algoritma untuk mengalokasikan slot waktu pada setiap *request* dalam penjadwalan *request* pada JRS topologi rantai dengan pendekatan *greedy*. Jika diketahui A adalah himpunan slot yang terurut naik dan bebas konflik jika dialokasikan pada kolom x_0, maka:

1. Nilai dari $D(i, x_0)$ dipilih dari slot yang bebas konflik pertama atau A(1).
2. $D(i, x_j + 1) = D(i, x_j) + 1$,
3. $D_i\_temp = [D(i, x_0), D(i, x_0) + 1, \dots, D(i, x_0) + k - 1]$.

memilih slot yang bebas konflik kedua atau A(2). Begitu seterusnya

hingga ditemukan D_i_temp yang bebas konflik.

5. Maka, $D_i = D_i\_temp$

Dengan menerapkan aturan ini, maka penjadwalan pada suatu urutan *request* JRS topologi rantai memenuhi syarat valid dan bebas konflik. Sedangkan untuk mencari urutan *request* yang dijadwalkan yang dapat menghasilkan jadwal dengan total slot waktu sekecil mungkin, pendekatan *greedy* yang digunakan adalah dengan cara mengurutkan kumpulan *request* dengan urutan berdasarkan panjang *path request* dan nomor dari simpul sumber.

Deskripsi Penjadwalan *Request* Pada JRS Topologi Rantai dengan Beberapa Jenis Pengurutan *Request*

Masalah 1: Diketahui sebuah jaringan dengan topologi rantai, banyak *vertex* =7 dan banyak *request* = 3, yaitu $r_1 = (1,4), r_2 = (3,6), dan\ r_3 = (7,3)$. *Request* r_1 dan r_2 adalah searah, sedangkan *request* r_3 berbeda arah dengan r_1 dan r_2. Maka solusi sementara masalah ini menggunakan algoritma *greedy* dengan jenis pengurutan: 1) terkiri dan terpanjang, 2) terpanjang dan terkiri, dan 3) terpendek dan terkanan diberikan pada Tabel 1.

Pada Tabel 1 nomor ke-3, menggunakan algoritma *greedy* dengan pengurutan *request* yang panjang *path* nya terpendek dan nomor simpul yang terbesar atau terkanan posisinya. Urutan *request* diperoleh dengan cara mengurutkan *request* dari yang terpendek lebih dulu. Jika terdapat beberapa *request* yang simpul sumbernya sama, urutkan *request* tersebut dari yang terkanan lebih dulu. Di antara $r_1 = (1,4),\ r_2 = (3,6),\ dan\ r_3 = (7,3)$ urutan *request* yang terkiri dan terpanjang adalah r_2, r_1, r_3. Penjadwalan untuk r_2, r_1, r_3 dilakukan dengan urutan seperti berikut: $D_1 = [d(r_2,3), d(r_2,4), d(r_2,5)]$, $D_2 = [d(r_1,1), d(r_1,2)d(r_1,3)]$ dan berikutnya $D_3 = [d(r_3,7), d(r_3,6), d(r_3,5),\ d(r_3,4)]$, dengan $d(r,x)$ adalah slot waktu untuk mengirimkan *request* r dari simpul x ke simpul berikutnya dalam fungsi routing r. Pada masalah ini, *request* r_1 dan r_2 searah sedangkan r_3 berbeda arah dengan r_1 dan r_2.

Langkah penjadwalan:

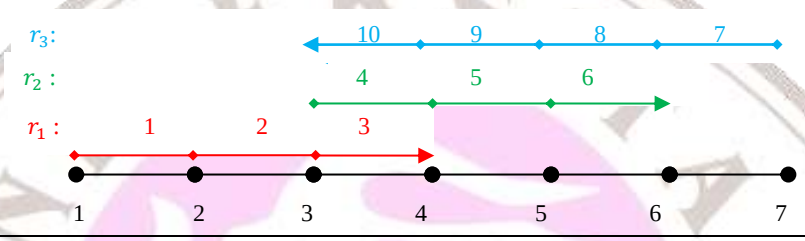
1. Jadwal *request* r_2 :$D_1 = [d(r_2,3), d(r_2,4), d(r_2,5)] = [1,2,3]$
2. Jadwal *request* r_1: $D_2 = [d(r_1,1), d(r_1,2)d(r_1,3)]$
   Untuk menentukan slot waktu $d(r_1,1)$, dimisalkan A adalah himpunan terurut naik dari slot waktu yang *available* jika dialokasikan pada kolom 1 (transmisi dari simpul 1 ke simpul 2).
   A= [{ 1,2,..,*max(AssignedSlot)*+1}-*ConflictSlot*]
   A= [{1,2,3,4}-{1}] = [2,3,4]
   Misalkan $d(r_1,1) = A(1) = 2$.
   Dengan menggunakan $d(r, x_{j+1}) = d(r, x_{j+1}) + 1$, diperoleh:
   D_2_temp
   $= [d(r_1,1), d(r_1,2)d(r_1,3)$
   $= [2,3,4]$
   $D_2\_temp[2,3,4]$ bebas konflik sehingga $D_2 = D_{2_{temp}} = [2,3,4]$
3. Jadwal *request* r_3: $D_3 = [d(r_3,7), d(r_3,6), d(r_3,5),\ d(r_3,4)]$
   Untuk menentukan slot waktu $d(r_3,7)$, dimisalkan A adalah himpunan terurut naik dari slot waktu yang *available* untuk dialokasikan pada kolom 6 (transmisi dari simpul 7 ke simpul 6).
   A= [{1,2,...,5}-{3}] = [1,2,4,5]

Misalkan $d(r_3,7) = A(1) = 1$
Dengan menggunakan $d(r, x_{j+1}) = d(r, x_{j+1}) + 1$, diperoleh :
D_3_temp
$= [d(r_3,7), d(r_3,6), d(r_3,5), d(r_3,4)]$
$= [1,2,3,4]$
$D_3\_temp = [1,2,3,4]$ konflik pada slot 2,3 dan 4 sehingga $D_3 \neq D_3\_temp$

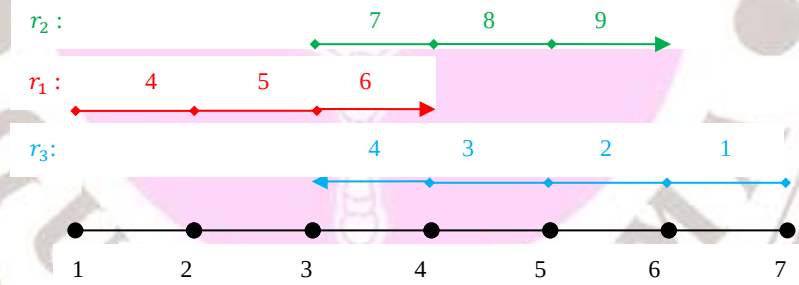
Misalkan $d(r_3, 7) = A(2) = 2$
Dengan menggunakan $d(r, x_{j+1}) = d(r, x_{j+1}) + 1$, diperoleh :
D_3_temp
$= [d(r_3, 7), d(r_3, 6), d(r_3, 5), d(r_3, 4)]$
$= [2,3,4,5]$
$D_3\_temp = [2,3,4,5]$ konflik pada slot 3,4 dan 5 sehingga $D_3 \neq D_3\_temp$
Misalkan $d(r_3, 7) = A(3) = 4$
Dengan menggunakan $d(r, x_{j+1}) = d(r, x_{j+1}) + 1$, diperoleh :
D_3_temp
$= [d(r_3, 7), d(r_3, 6), d(r_3, 5), d(r_3, 4)]$
$= [4,5,6,7]$
$D_3\_temp = [4,5,6,7]$ bebas konflik sehingga $D_3 = D_{3_{temp}} = [4,5,6,7]$.

Tabel 1. Beberapa Solusi Sementara Masalah 1

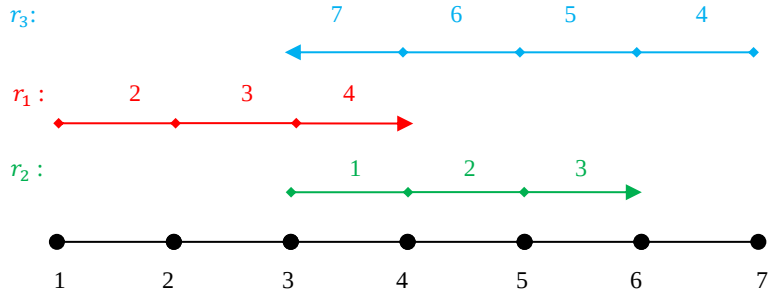
1. Menggunakan Pengurutan *request* yang terkiri dan terpanjang

Urutan *request*	Ilustrasi Jadwal	max (d)
r_1, r_2, r_3		10

2. Menggunakan pengurutan *request* yang terpanjang dan terkiri

Urutan *request*	Ilustrasi Jadwal	max (d)
r_3, r_1, r_2		9

3. Menggunakan pengurutan *request* yang terpendek dan terkanan

Urutan *request*	Ilustrasi Jadwal	max (d)
r_2, r_1, r_3		7

Dari penjadwalan ini diperoleh total slot waktu 7. Dengan cara yang sama, algoritma *greedy* dengan pengurutan terkiri dan terpanjang menghasilkan urutan *request* r_1, r_2, r_3 dan total slot waktu 10. Kemudian, algoritma *greedy* dengan pengurutan terpanjang dan terkiri menghasilkan urutan *request* r_3, r_1, r_2 dan total slot waktu 9. Solusi terbaik dari 3 solusi sementara adalah 7 dengan urutan *request* terpendek dan terkanan.

Untuk selanjutnya dipelajari 22 jenis pengurutan dengan pendekatan *greedy* dalam mengalokasikan slot waktu untuk setiap *request* pada penjadwalan *request* JRS topologi rantai.

Implementasi Algoritma *Greedy* untuk data DawnC01

Data DawnC01 terdiri atas kumpulan *request* sebanyak 10 *request* (m) pada JRS topologi rantai yang terdiri atas 10 *node* (n), dengan simpul sumber (s) dan simpul tujuan (t) untuk setiap *request* diberikan oleh Tabel 2.

Tabel 2. Kumpulan *Request* DawnC01

Kumpulan *request*		
No	s	t
1	9	2
2	8	5
3	7	4
4	5	8
5	10	9
6	7	10
7	7	8
8	9	3
9	2	9
10	5	8

Dari hasil dari penjadwalan menggunakan algoritma greedy untuk data DawanC01 tanpa pengurutan (trivial) diperoleh total slot waktu 26.

Sedangkan implementasi algoritma *greedy* dengan 22 jenis pengurutan untuk data DawnC01 menghasilkan 22 solusi sementara, dengan urutan *request* dan total slot waktu untuk setiap jenis pengurutan dapat dilihat pada Tabel 3.

Dari Tabel 3 dapat disimpulkan bahwa solusi algoritma *greedy* untuk data DawnC01 adalah solusi sementara yang menghasilkan total slot waktu terkecil, yaitu 21. Pada kasus ini solusi yang menghasilkan total slot waktu 21, dipenuhi oleh jenis pengurutan ke-4, 10, 13, dan 19. Pada Tabel 4 diberikan solusi algoritma *greedy* dengan pengurutan jenis ke-19, yaitu *request* terkanan dan terpanjang untuk data DawnC01 berupa urutan *request* yang dijadwalkan, penjadwalan dan total slot waktu.

Pada Tabel 5 diberikan perbandingan hasil dari penjadwalan menggunakan algoritma tanpa pengurutan *request* (trivial) dan algoritma dengan pengurutan (*greedy*) untuk beberapa data dengan banyak simpul (n) dan banyak *request* (m) yang bervariasi.

Tabel 3. Urutan *Request* dan Total Slot Waktu dari 22 Solusi Sementara Algoritma *Greedy* untuk Data DawnC01

Jenis pengurutan *greedy*	Urutan *Request* untuk 22 jenis pengurutan										Total slot waktu
1	9	4	10	6	7	5	3	2	8	1	**25**
2	1	8	3	2	5	7	4	10	6	9	**23**
3	9	4	10	3	7	6	2	1	8	5	**24**
4	5	1	8	2	3	7	6	4	10	9	**21**
5	9	6	4	10	7	5	2	3	8	1	**27**
6	1	8	2	3	5	7	6	4	10	9	**23**
7	5	8	1	2	6	7	3	4	10	9	**24**
8	9	4	10	3	7	6	2	1	8	5	**24**
9	9	4	10	3	6	7	2	1	8	5	**25**
10	5	1	8	2	3	6	7	4	10	9	**21**
11	9	4	6	10	7	5	2	3	8	1	**26**
12	1	8	2	3	5	7	4	6	10	9	**23**
13	9	1	8	4	10	3	6	2	7	5	**21**
14	7	5	4	10	3	6	2	8	9	1	**32**
15	9	4	10	7	3	6	2	8	1	5	**25**
16	5	8	1	2	7	3	6	4	10	9	**24**
17	1	9	8	2	3	6	4	10	5	7	**22**
18	5	7	2	3	6	4	10	8	1	9	**30**
19	5	1	8	2	3	6	7	4	10	9	**21**
20	9	4	10	7	3	6	2	8	1	5	**25**
21	1	9	8	2	3	4	6	10	5	7	**22**
22	5	7	2	3	4	6	10	8	1	9	**30**

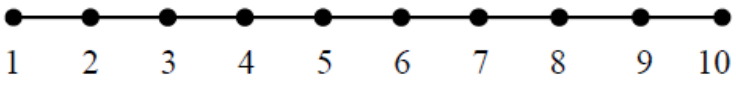
Tabel 4. Solusi Penjadwalan DawnC01

Urutan *request*		
No	s	T
5	10	9
1	9	2
8	9	3
2	8	5
3	7	4
6	7	10
7	7	8
4	5	8
10	5	8
9	2	9

Penjadwalan DawnC01

total slot waktu = 21

								1
	8	7	6	5	4	3	2	
		10	9	8	7	6	5	
				11	10	9		
			3	2	1			
						11	12	13
						8		
				12	13	14		
				15	16	17		
	15	16	17	18	19	20	**21**	



Tabel 5. Total Slot Waktu Hasil Algoritma Trivial dan Algoritma *Greedy* untuk Data DawnC01-DawnC08

n	m	Data	Total Slot Waktu	
			Trivial	*Greedy*
10	10	DawnC01	26	21
10	20	DawnC02	35	32
10	50	DawnC03	90	71
20	10	DawnC04	35	21
20	20	DawnC05	43	34
30	20	DawnC06	65	47
50	30	DawnC07	149	92
100	20	DawnC08	206	124

SIMPULAN DAN SARAN

Pada penelitian ini telah dibuat algoritma penjadwalan yang dapat menyelesaikan dua bagian masalah penjadwalan pada JRS topologi rantai, yaitu bagaimana menentukan urutan *request* dalam mengelokasikan slot waktu dan bagaimana teknik meng-alokasikan slot waktu pada setiap *re-quest* yang dapat menghasilkan penjad-walan dengan total slot waktu minimal. Untuk 8 data (DawnC01 – DawnC08) diperoleh bahwa algoritma *greedy* menghasilkan jadwal yang lebih baik (total slot waktu lebih sedikit) daripada algoritma tanpa pengurutan *request* (trivial), dengan peningkatan rata-rata 10,18 %.

DAFTAR PUSTAKA

[1] [Darties, et al, 2008] Darties, B., Sylvain, D., dan Jerome, P. 2008. *Request Satisfication Problem in Synchronous Radio Network.* ADHOC-NOW'08: 7th International Conference Ad-Hoc, Mobile and Wireless Networks, Sophia Antipolis, Nice, France.

[2] [Alon, et al, 1991] Alon, N., Bar-Noy, A., Linial, N., dan Peleg, D. 1991. "A Lower bound for radio broadcast". *J. Comput. Syst. Sci.*, Vol. 43, No. 2, pp. 290 – 298.

[3] [Ephemides dan Truong, 1990] Ephemides, A. dan Truong, T. 1990. "Scheduling broadcast in multihop radio networks". *IEEE Transactions on Communications*, Vol. 38, pp. 456 – 461.

[4] [Ramanathan dan Lloyd, 1993] Ramanathan, S. dan Lloyd, E., 1993. "Scheduling Algorithm fo Multihop Radio Networks". *IEEE/ACM Transactions on Networking*, Vol. 1, pp. 166 – 177.

[5] [Garey dan Johnson, 1979] Garey, M. R. dan Johnson, D. S. 1979. Computer and Intractability: A guide to the theory of NP-completeness. Freeman.